Reconfiguration and message losses in parameterized broadcast networks

Nathalie Bertrand, Patricia Bouyer, Anirban Majumdar

Reconfiguration and message losses in parameterized broadcast networks

Nathalie Bertrand
Univ. Rennes, Inria, CNRS, IRISA – Rennes (France)

Patricia Bouyer
LSV, CNRS & ENS Paris-Saclay, Univ. Paris-Saclay – Cachan (France)

Anirban Majumdar
Univ. Rennes, Inria, CNRS, IRISA – Rennes (France)
LSV, CNRS & ENS Paris-Saclay, Univ. Paris-Saclay – Cachan (France)

Abstract

Broadcast networks allow one to model networks of identical nodes communicating through message broadcasts. Their parameterized verification aims at proving a property holds for any number of nodes, under any communication topology, and on all possible executions. We focus on the coverability problem which dually asks whether there exists an execution that visits a configuration exhibiting some given state of the broadcast protocol. Coverability is known to be undecidable for static networks, *i.e.* when the number of nodes and communication topology is fixed along executions. In contrast, it is decidable in PTIME when the communication topology may change arbitrarily along executions, that is for reconfigurable networks. Surprisingly, no lower nor upper bounds on the minimal number of nodes, or the minimal length of covering execution in reconfigurable networks, appear in the literature.

In this paper we show tight bounds for cutoff and length, which happen to be linear and quadratic, respectively, in the number of states of the protocol. We also introduce an intermediary model with static communication topology and non-deterministic message losses upon sending. We show that the same tight bounds apply to lossy networks, although, reconfigurable executions may be linearly more succinct than lossy executions. Finally, we show NP-completeness for the natural optimisation problem associated with the cutoff.

2012 ACM Subject Classification Theory of computation → Verification by model checking

Keywords and phrases model checking – parameterized verification – broadcast networks

Funding The second author was supported by ERC project EQualIS (308087).

1 Introduction

Parameterized verification. Systems formed of many identical agents arise in many concrete areas: distributed algorithms, populations, communication or cache-coherence protocols, chemical reactions etc. Models for such systems depend on the communication or interaction means between the agents. For example pairwise interactions are commonly used for populations of individuals, whereas selective broadcast communications are more relevant for communication protocols on ad-hoc networks. The capacity of the agents, and thus models that are used to represent their behaviour also vary.

Verifying such systems amounts to checking that a property holds independently of the number of agents. Typically, a consensus algorithm should be correct for any number of participants. We refer to these systems as parameterized systems, and the parameter is the number of agents. The verification of parameterized systems started in the late 80's and recently regained attention from the model-checking community [11, 8, 6, 1]. It can be seen as particular cases of infinite-state-system verification, and the fact that all agents are identical can sometimes lead to efficient algorithms [5].

Broadcast networks. This paper targets the application to protocols over ad-hoc networks, and we thus focus on the model of broadcast networks [3]. A broadcast network is composed of several nodes that execute the same broadcast protocol. The latter is a finite automaton, where transitions are labeled with message sendings or message receptions. Configuration in broadcast networks is then comprised of a set of agents, their current local states, together with a communication topology (which represents which agents are within radio range). A transition represents the effect of one agent sending a message to its neighbours.

Parameterized verification of broadcast networks amounts to checking a given property independently of the initial configuration, and in particular independently of the number of agents and communication topology. A natural property one can be interested in is coverability: a state of the broadcast protocol is coverable if some execution leads to a configuration in which one node is in that local state. When considering error states, a positive instance for the coverability problem thus corresponds to a network that can exhibit a bad behaviour.

Coverability is undecidable for static broadcast networks [3], *i.e.* when the communication topology is fixed along executions. Decidability can be recovered by relaxing the semantics and allowing non-deterministic reconfigurations of the communication topology. In reconfigurable broadcast networks, coverability of a control state is decidable in PTIME [2]. A simple saturation algorithm allows to compute the set of all states of the broadcast protocol that can be covered.

Cutoff and covering length. Two important characteristics of positive instances of the coverability problem are the cutoff and the covering length. First, the *cutoff* is the minimal number of agents for which a covering execution exists. The notion of cutoff is particularly relevant for reconfigurable broadcast networks since they enjoy a monotonicity property: if a state can be covered from a configuration, it can also be from any configuration with more nodes. Second, the *covering length* is the minimal number of steps for covering executions. It weighs how fast a network execution can go wrong. Both the cutoff and the covering length are somehow complexity measures for the coverability problem. Surprisingly, no upper nor lower bounds on these values appear in the literature for reconfigurable broadcast networks.

Contributions. In this paper, we prove a tight linear bound for the cutoff, and a tight quadratic bound for the covering length in reconfigurable broadcast networks. Both are expressed in the number of states of the broadcast protocol. These are obtained by refining the saturation algorithm that computes the set of coverable states, and finely analysing it.

Another contribution is to introduce lossy broadcast networks, in which the communication topology is fixed, however errors in message transmission may occur. In contrast with broadcast networks with losses that appear in the literature [4], in our model, message losses happen upon sending, rather than upon reception. This makes a crucial difference: reconfiguration of the communication topology can easily be encoded by losses upon reception, whereas it is not obvious for losses upon sending. Perhaps surprisingly, we prove that the set of states that can be covered in reconfigurable semantics agrees with the one in static lossy semantics. Using the same refined saturation algorithm, we prove that same tight bounds hold for lossy broadcast networks: the cutoff is linear, and the covering length is quadratic (in the number of states of the broadcast protocol). The two semantics thus appear quite similar, yet, we show that the reconfigurable semantics can be linearly more succinct (in terms of number of nodes) than the lossy semantics.

Finally, we study a natural decision problem related to the cutoff: decide whether a state is coverable (in either semantics) with a fixed number of nodes. We prove it to be NP-complete.

Outline. In Section 2, we define the broadcast networks, with static, reconfiurable and lossy semantics. In Section 3, we present our tight bounds for cutoff and covering length. In Section 4, we show our succinctness result. In Section 5, we give our NP-completeness result.

2 Broadcast networks

2.1 Static broadcast networks

▶ **Definition 1.** *A* broadcast protocol *is a tuple* $\mathcal{P} = (Q, I, \Sigma, \Delta)$ *where* Q *is a finite set of control states;* $I \subseteq Q$ *is the set of initial control states;* Σ *is a finite alphabet; and* $\Delta \subseteq (Q \times \{!a, ?a \mid a \in \Sigma\} \times Q)$ *is the transition relation.*

For ease of readability, we often write $q \xrightarrow{!a} q'$ (resp. $q \xrightarrow{?a} q'$) for $(q, !a, q') \in \Delta$ (resp. $(q, ?a, q') \in \Delta$). We assume all broadcast networks to be complete for receptions: for every $q \in Q$ and $a \in \Sigma$, there exists q' such that $q \xrightarrow{?a} q'$.

A broadcast protocol is represented in Figure 1. In this example and in the whole paper, for concision purposes, we assume that if the reception of a message is unspecified from some state, it implicitly represents a self-loop. For example here, from q_1, receiving a leads to q_1 again.

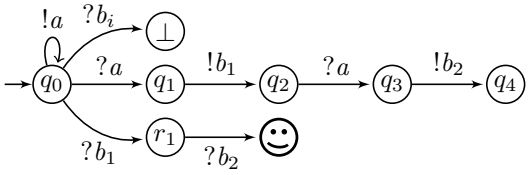

Figure 1 Example of a broadcast protocol.

Broadcast networks involve several copies, or *nodes*, of the same broadcast protocol $\mathcal{P}$. A configuration is an undirected graph whose vertices are labelled with a state of Q. Transitions between configurations happen by broadcasts from a node to its neighbours.

Formally, given a broadcast protocol $\mathcal{P} = (Q, I, \Sigma, \Delta)$, a *configuration* is an undirected graph $\gamma = (\mathsf{N}, \mathsf{E}, \mathsf{L})$ where N is a finite set of nodes; $\mathsf{E} \subseteq \mathsf{N} \times \mathsf{N}$ is a symmetric and irreflexive relation describing the set of edges; finally, $\mathsf{L} \colon \mathsf{N} \to Q$ is the labelling function. We let $\Gamma(\mathcal{P})$ denote the (infinite) set of Q-labelled graphs. Given a configuration $\gamma \in \Gamma(\mathcal{P})$, we write $\mathsf{n} \sim \mathsf{n}'$ whenever $(\mathsf{n}, \mathsf{n}') \in \mathsf{E}$ and we let $\mathsf{Neigh}_\gamma(\mathsf{n}) = \{\mathsf{n}' \in \mathsf{N} \mid \mathsf{n} \sim \mathsf{n}'\}$ be the neighbourhood of n, *i.e.* the set of nodes adjacent to n. Finally $\mathsf{L}(\gamma)$ denotes the set of labels appearing in nodes of γ. A configuration $(\mathsf{N}, \mathsf{E}, \mathsf{L})$ is called *initial* if $\mathsf{L}(\mathsf{N}) \subseteq I$.

The operational semantics of a static broadcast network for a given broadcast protocol $\mathcal{P}$ is an infinite-state transition system $\mathcal{T}(\mathcal{P})$. Intuitively, each node of a configuration runs protocol $\mathcal{P}$, and may send/receive messages to/from its neighbours. From a configuration $\gamma = (\mathsf{N}, \mathsf{E}, \mathsf{L})$, there is a step to $\gamma' = (\mathsf{N}', \mathsf{E}', \mathsf{L}')$ if $\mathsf{N}' = \mathsf{N}$, $\mathsf{E}' = \mathsf{E}$, and there exists $\mathsf{n} \in \mathsf{N}$ and $a \in \Sigma$ such that $(\mathsf{L}(\mathsf{n}), !a, \mathsf{L}'(\mathsf{n})) \in \Delta$, and for every $\mathsf{n}' \in \mathsf{N}$, if $\mathsf{n}' \in \mathsf{Neigh}_\gamma(\mathsf{n})$, then $(\mathsf{L}(\mathsf{n}'), ?a, \mathsf{L}'(\mathsf{n}')) \in \Delta$, otherwise $\mathsf{L}'(\mathsf{n}') = \mathsf{L}(\mathsf{n}')$: a step reflects how nodes evolve when one of them broadcasts a message to its neighbours. We write $\gamma \xrightarrow{\mathsf{n},!a}_{\mathsf{s}} \gamma'$, or simply $\gamma \to_{\mathsf{s}} \gamma'$ (the s subscript emphasizes that the communication topology is *static*).

An *execution* of the static broadcast network is a sequence $\rho = (\gamma_i)_{0 \le i \le r}$ of configurations $(\mathsf{N}, \mathsf{E}, \mathsf{L}_i)$ such that γ_0 is an initial configuration, and for every $0 \le i < r$, $\gamma_i \to_{\mathsf{s}} \gamma_{i+1}$. We

write $\#\mathsf{nodes}(\rho)$ for the number of nodes in γ_0, $\#\mathsf{steps}(\rho)$ for the number r of steps along ρ, and for any node $\mathsf{n} \in \mathsf{N}, \#\mathsf{steps}(\rho, \mathsf{n})$ for the number of broadcasts, called the *active length*, of node n along ρ. Note that, along an execution, the number of nodes and the communication topology are fixed. The set of all static executions is denoted $\mathsf{Exec}_\mathtt{s}(\mathcal{P})$.

Coverability problem.

Given a broadcast protocol $\mathcal{P}$ and a subset of target states $F \subseteq Q$, we write $\mathsf{COVER}_\mathtt{s}(\mathcal{P}, F)$ for the set of all *covering* executions, that is, executions that visit a configuration with a node labelled by a state in F:

$$\mathsf{COVER}_\mathtt{s}(\mathcal{P}, F) = \{(\gamma_i)_{0 \leq i \leq r} \in \mathsf{Exec}_\mathtt{s}(\mathcal{P}) \mid \mathsf{L}(\gamma_r) \cap F \neq \emptyset\}.$$

The *coverability problem* is a decision problem that takes a broadcast protocol $\mathcal{P}$ and a subset of target states F as inputs, and outputs whether $\mathsf{COVER}_\mathtt{s}(\mathcal{P}, F)$ is nonempty. For broadcast networks, the coverability problem is a parameterized verification problem, since the number of initial configurations is infinite. It is known that coverability is undecidable for static broadcast networks [3], since one can use the communication topology to build chains that may encode values of counters, and hence simulate Minsky machines [10].

If the broadcast protocol $\mathcal{P}$ allows to cover the subset F, we define the *cutoff* as the minimal number of nodes required in an execution to cover F. Similarly, we define the *covering length* as the length of a shortest finite execution covering F. Those values are important to characterize the complexity of a broadcast protocol: assuming a safe set of states, coverability of the complement set represents bad behaviours, and cutoff and covering length measure the size of minimal witnesses for violation of the safety property.

2.2 Reconfigurable broadcast networks

To circumvent the undecidability of coverability for static broadcast networks, one attempt is to introduce non-deterministic reconfiguration of the communication topology. This solution also allows one to model arbitrary mobility of the nodes, which is meaningful, *e.g.* for mobile ad-hoc networks [3].

Under this semantics, configurations are the same as under the static semantics. Transitions between configurations however are enhanced by the ability to modify the communication topology before performing a broadcast. Formally, from a configuration $\gamma = (\mathsf{N}, \mathsf{E}, \mathsf{L})$, there is a step to $\gamma' = (\mathsf{N}', \mathsf{E}', \mathsf{L}')$ if $\mathsf{N}' = \mathsf{N}$, and there exists $\mathsf{n} \in \mathsf{N}$ and $a \in \Sigma$ such that $(\mathsf{L}(\mathsf{n}), !a, \mathsf{L}'(\mathsf{n})) \in \Delta$, and for every $\mathsf{n}' \in \mathsf{N}$, if $\mathsf{n}' \in \mathsf{Neigh}_{\gamma'}(\mathsf{n})$, then $(\mathsf{L}(\mathsf{n}), ?a, \mathsf{L}'(\mathsf{n}')) \in \Delta$, otherwise $\mathsf{L}'(\mathsf{n}') = \mathsf{L}(\mathsf{n}')$: a step thus reflects that the communication topology may change from E to E' followed by the broadcast of a message from a node to its neighbours in the new topology. We write $\gamma \xrightarrow{\mathsf{n}, !a}_\mathtt{r} \gamma'$, or simply $\gamma \to_\mathtt{r} \gamma'$.

Similarly to the static case, we write $\mathsf{Exec}_\mathtt{r}(\mathcal{P})$ and $\mathsf{COVER}_\mathtt{r}(\mathcal{P}, F)$ for, respectively the set of all reconfigurable executions in $\mathcal{P}$, and the set of all reconfigurable executions in $\mathcal{P}$ that cover F. We will also use the same notations $\#\mathsf{nodes}(\rho)$, $\#\mathsf{steps}(\rho)$ and $\#\mathsf{steps}(\rho, \mathsf{n})$ as in the static case.

Figure 7 (with $n = 2$) gives an example of reconfigurable execution for the broadcast protocol of Figure 1 (which covers ☺). Note that the communication topology indeed evolves along the execution. Here the colored nodes broadcast a message in the step leading to the next configuration.

A noticeable property of reconfigurable broadcast networks is the following copycat property. Such a monotonicity property was originally shown in [7] for asynchronous shared-memory systems, and it also applies to our context.

▶ **Proposition 2** (Copycat for reconfigurable semantics). *Given* $\rho : \gamma_0 \to_{\mathtt{r}} \gamma_1 \cdots \to_{\mathtt{r}} \gamma_s$ *an execution, with* $\gamma_s = (\mathsf{N}, \mathsf{E}, \mathsf{L})$*, for every* $q \in \mathsf{L}(\gamma_s)$*, for every* $\mathsf{n}^q \in \mathsf{N}$ *such that* $\mathsf{L}(\mathsf{n}^q) = q$*, there exists* $t \in \mathbb{N}$ *and an execution* $\rho' : \gamma'_0 \to_{\mathtt{r}} \gamma'_1 \cdots \to_{\mathtt{r}} \gamma'_t$ *with* $\gamma'_t = (\mathsf{N}', \mathsf{E}', \mathsf{L}')$ *such that* $|\mathsf{N}'| = |\mathsf{N}|+1$*, there is an injection* $\iota : \mathsf{N} \to \mathsf{N}'$ *with for every* $\mathsf{n} \in \mathsf{N}$*,* $\mathsf{L}'(\iota(\mathsf{n})) = \mathsf{L}(\mathsf{n})$*, and for the extra node* $\mathsf{n}_{\mathsf{fresh}} \in \mathsf{N}' \setminus \iota(\mathsf{N})$*,* $\mathsf{L}'(\mathsf{n}_{\mathsf{fresh}}) = q$*, and* $\#\mathit{steps}(\rho', \mathsf{n}_{\mathsf{fresh}}) = \#\mathit{steps}(\rho, \mathsf{n}^q)$.

Intuitively, the new node $\mathsf{n}_{\mathsf{fresh}}$ will copy the moves of node n^q: it performs the same broadcasts (but to nobody) and receives the same messages. More precisely, when n^q broadcasts in ρ, it does so also in ρ' and then we disconnect all the nodes and $\mathsf{n}_{\mathsf{fresh}}$ repeats the broadcast (no other node is affected because of the disconnection); when n^q receives a message in ρ, we connect $\mathsf{n}_{\mathsf{fresh}}$ to the same neighbours as n^q (*i.e.*, $\iota(\mathsf{n}) \sim' \mathsf{n}_{\mathsf{fresh}}$ if and only if $\mathsf{n} \sim \mathsf{n}^q$) so that $\mathsf{n}_{\mathsf{fresh}}$ also receives the same message in ρ'.

Relying on the copycat property, when reconfigurations are allowed, the coverability problem becomes decidable and solvable in polynomial time.

▶ **Theorem 3** ([2]). *Coverability is decidable in* PTIME *for reconfigurable broadcast networks.*

More precisely, a simple saturation algorithm allows one to compute in polynomial time, the set of all states that can be covered. Despite this complexity result, to the best of our knowledge, no bounds on the cutoff or length of witness executions are stated in the literature.

2.3 Broadcast networks with messages losses

Communication failures were studied for broadcast networks, assuming non-deterministic message losses could happen: when a message is broadcast, some of the neighbours of the sending node may not receive it [4]. As observed by the authors, the coverability problem for such networks easily reduces to the coverability problem in reconfigurable networks by considering a complete topology, and message losses are simulated by reconfigurations. Thus, message losses upon reception are equivalent to reconfiguration of the communication topology.

We propose an alternative semantics here: when a message is broadcast, it either reaches all neighbours of the sending node, or none of them. This is relevant in contexts where broadcasts are performed in an atomic manner and may fail. In contrast to message losses upon reception, it is not obvious to simulate arbitrary reconfigurations of the communication topology with such message losses.

Formally, from a configuration $\gamma = (\mathsf{N}, \mathsf{E}, \mathsf{L})$, there is a step to $\gamma' = (\mathsf{N}', \mathsf{E}', \mathsf{L}')$ if $\mathsf{N}' = \mathsf{N}$, $\mathsf{E}' = \mathsf{E}$ and there exists $\mathsf{n} \in \mathsf{N}$ and $a \in \Sigma$ such that $(\mathsf{L}(\mathsf{n}), !a, \mathsf{L}'(\mathsf{n})) \in \Delta$, and either (a) for every $\mathsf{n}' \neq \mathsf{n}$, $\mathsf{L}'(\mathsf{n}') = \mathsf{L}(\mathsf{n}')$ (no one has received the message, it has been lost), or (b) if $\mathsf{n}' \in \mathsf{Neigh}_{\gamma'}(\mathsf{n})$, then $(\mathsf{L}(\mathsf{n}'), ?a, \mathsf{L}'(\mathsf{n}')) \in \Delta$, otherwise $\mathsf{L}'(\mathsf{n}') = \mathsf{L}(\mathsf{n}')$: a step thus reflects that the broadcast message may be lost when it is sent. We write $\gamma \xrightarrow{\mathsf{n}.!a}_1 \gamma'$ or simply $\gamma \to_1 \gamma'$. Similarly to the static and reconfigurable semantics, $\#\mathsf{steps}(\rho, \mathsf{n})$ is the number of broadcasts (including lost ones) by node n along ρ; and we write $\#\mathsf{nonlost\_steps}(\rho, \mathsf{n})$ for the number of successful broadcasts by node n along ρ.

For lossy executions also, we use the following notations: $\mathsf{Exec}_1(\mathcal{P})$ and $\mathsf{COVER}_1(\mathcal{P}, F)$. Any lossy execution can be seen as a reconfigurable execution. Indeed, a lossy execution

with communication topology E can be transformed into a reconfigurable one in which the communication topology of each configuration is either $\emptyset$ or E, depending on whether the next broadcast is lost or not. Therefore, with slight abuse of notation, we write $\mathsf{Exec}_{\mathtt{l}}(\mathcal{P}) \subseteq \mathsf{Exec}_{\mathtt{r}}(\mathcal{P})$.

Figure 2 gives an example of a lossy execution for the broadcast protocol of Figure 1. Note that in the third transition, some node indeed performs a lossy broadcast, emphasized by the subscript "lost". As before, the colored nodes broadcast a message in the step leading to the next configuration.

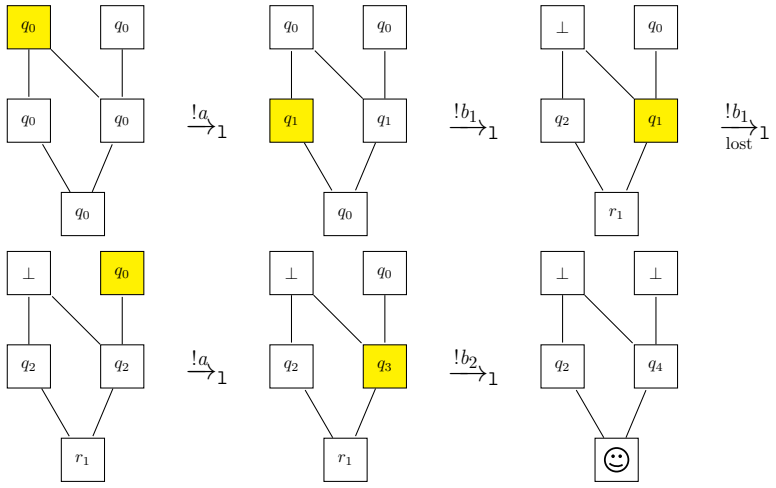

Figure 2 Example of a lossy execution on the protocol from Figure 1.

3 Tight bounds for reconfigurable and lossy broadcast networks

In this section, we will show tight bounds for the cutoff and the minimal length of a witness execution for the coverability problem. These hold both for the reconfigurable and the lossy semantics.

3.1 Upper bounds on cutoff and covering length for reconfigurable networks

First, we will refine the polytime saturation algorithm of [2], which computes all states which can be covered in the reconfigurable semantics. We will then show that, based on the underlying computation, one can construct small witnesses for the two semantics (linear number of nodes and quadratic number of steps). While it would be enough to show the result for the lossy semantics (since, given a broadcast protocol $\mathcal{P}$, $\mathsf{Exec}_{\mathtt{l}}(\mathcal{P}) \subseteq \mathsf{Exec}_{\mathtt{r}}(\mathcal{P})$), for pedagogical reasons, we provide the two proofs, starting with the simplest one for reconfigurable semantics.

Let us fix for the rest of this section, a protocol $\mathcal{P} = (Q, I, \Sigma, \Delta)$. We slightly modify the algorithm given in [2] as follows: we include at most one state to the set S in each iteration. Additionally, we associate a labelling function $c : S \to \mathbb{N}$ with the set S in every iteration. More formally, we consider the modification of the previous saturation algorithm as shown in Algorithm 1.

Algorithm 1 Refined saturation algorithm for coverability

$S := I$; $c(S) := |I|$; $S' := \emptyset$
while $S \neq S'$ **do**
  $S' := S$; $c := c(S)$
  if $\exists (q_1, !a, q_2) \in \Delta$ s.t. $q_1 \in S'$ and $q_2 \notin S'$ **then**
    $S := S \cup \{q_2\}$; $c(S) := c + 1$
  else if $\exists (q_1, !a, q_2) \in \Delta$ and $(q_1', ?a, q_2') \in \Delta$ s.t. $q_1, q_2, q_1' \in S'$ and $q_2' \notin S'$ **then**
    $S := S \cup \{q_2'\}$; $c(S) := c + 2$
  end if
end while
return S

In Algorithm 1, the variable c counts the number of nodes that are sufficient to cover the current set S, as we will prove later.

▶ **Lemma 4** ([2]). *Algorithm 1 terminates and returns the set of coverable states. In particular,* $\mathsf{COVER}_{\mathtt{r}}(\mathcal{P}, F) \neq \emptyset$ *iff* $F \cap S \neq \emptyset$.

Let $S_0, S_1, \ldots, S_m$ be the sets after each iteration of the algorithm, with $S_0 = I$ and $S_m = S$. We fix an ordering on the states in S on the basis of insertion in S: for all $1 \leq i \leq m$, q_i is such that $q_i \in S_i \setminus S_{i-1}$. In the following, we show the desired upper bounds, proving that there exists an execution of size $O(n)$ and length $O(n^2)$ covering at the same time all states of S_m.

▶ **Theorem 5.** *Let* $\mathcal{P} = (Q, I, \Sigma, \Delta)$ *be a broadcast protocol, and* $F \subseteq Q$. *If* $\mathsf{COVER}_{\mathtt{r}}(\mathcal{P}, F) \neq \emptyset$ *(that is, if* $F \cap S \neq \emptyset$*), then there exists* $\rho \in \mathsf{COVER}_{\mathtt{r}}(\mathcal{P}, F)$ *with* $\#\mathsf{nodes}(\rho) \leq 2|Q|$ *and* $\#\mathsf{steps}(\rho) \leq 2|Q|^2$.

Theorem 5 is a consequence of the following Lemma.

▶ **Lemma 6.** *For every step* i *of Algorithm 1, there exists an initial configuration* γ_0, *a configuration* γ *and a reconfigurable execution* $\rho : \gamma_0 \xrightarrow{*}_{\mathtt{r}} \gamma$ *such that* $\mathsf{L}(\gamma) = S_i$, $\#\mathsf{nodes}(\rho) = c(S_i)$, *and* $\max_{\mathsf{n}} \#\mathsf{steps}(\rho, \mathsf{n}) \leq i$.

Proof. The lemma is proved by induction on i. The base case $i = 0$ is obvious: take the initial configuration γ_0 with $|I|$ nodes, and label each node with a different initial state; its size is $|I|$, and the length of the execution is 0, hence so is the maximum active length.

To prove the induction step, we distinguish two cases: depending on whether q_{i+1} was added as the target state of a broadcast transition $q \xrightarrow{!a}$ for some $q \in S_i$; or whether q_{i+1} is the target state of a reception from some $q \in S_i$ with matching broadcast between two states already in S_i.

Case 1: There exists $q \in S_i$ with $q \xrightarrow{!a} q_{i+1}$. We apply the induction hypothesis to step i, and exhibit an execution $\rho : \gamma_0 \xrightarrow{*}_{\mathtt{r}} \gamma$ such that $\mathsf{L}(\gamma) = S_i$, $\#\mathsf{nodes}(\rho) = c(S_i)$ and $\max_{\mathsf{n}} \#\mathsf{steps}(\rho, \mathsf{n}) \leq i$. Applying the copycat property (see Proposition 2), we construct an execution $\rho' : \gamma_0' \xrightarrow{*}_{\mathtt{r}} \gamma'$ such that γ_0' has one node more than γ_0, and, focusing on the nodes (since we are in a reconfigurable setting, edges in the configuration are not important), γ' coincides with γ, with an extra node n labelled by q. We then disconnect all nodes and extend with a transition $\gamma' \xrightarrow{\mathsf{n},!a}_{\mathtt{r}} \gamma''$, which makes only progress node n from q to q_{i+1}; the resulting execution is denoted ρ''. Then:

1. $\mathsf{L}(\gamma'') = S_i \cup \{q_{i+1}\} = S_{i+1}$,

2. $\#\mathsf{nodes}(\rho'') = c(S_i) + 1 = c(S_{i+1})$,
3. $\max_{\mathsf{n}} \#\mathsf{steps}(\rho'', \mathsf{n}) \leq \max_{\mathsf{n}} \#\mathsf{steps}(\rho, \mathsf{n}) + 1 \leq i + 1$; Indeed, the active length of the copycat node along ρ' coincides with the active length of some existing node along ρ, and it is increased only by 1 in ρ''.

This proves the induction step in the first case.

Case 2: There exists $q, q', q'' \in S_i$ with $q \xrightarrow{?a} q_{i+1}$ and $q' \xrightarrow{!a} q''$. The idea is similar to the previous case, but one should apply the copycat property twice, to both q and q'. We formalize this.

We apply the induction hypothesis to step i, and exhibit an execution $\rho : \gamma_0 \xrightarrow{*}_{\mathtt{r}} \gamma$ such that $\mathsf{L}(\gamma) = S_i$, $\#\mathsf{nodes}(\rho) = c(S_i)$ and $\max_{\mathsf{n}} \#\mathsf{steps}(\rho, \mathsf{n}) \leq i$. Applying the copycat property (see Proposition 2) twice, to both q and q', we construct an execution $\rho' : \gamma'_0 \xrightarrow{*}_{\mathtt{r}} \gamma'$ such that γ'_0 has two nodes more than γ_0, and, focusing on the nodes, γ' coincides with γ, with one extra node n labelled by q and one extra node n' labelled by q'. We then connect nodes n and n' and disconnect all other nodes, and extend with a transition $\gamma' \xrightarrow{\mathsf{n}',!a}_{\mathtt{r}} \gamma''$; this makes node n progress from q to q_{i+1} and node n' progress from q' to q''; all other nodes are unchanged; the resulting execution is denoted ρ''. Then:

1. $\mathsf{L}(\gamma'') = S_i \cup \{q'', q_{i+1}\} = S_{i+1}$ since $q'' \in S_i$,
2. $\#\mathsf{nodes}(\rho'') = c(S_i) + 2 = c(S_{i+1})$,
3. $\max_{\mathsf{n}} \#\mathsf{steps}(\rho'', \mathsf{n}) \leq \max_{\mathsf{n}} \#\mathsf{steps}(\rho, \mathsf{n}) + 1 \leq i + 1$; Indeed the active length of any of the copycat node along ρ' coincides with the active length of some existing node along ρ, and it is increased by at most 1 in ρ''.

This proves the induction step in the second case, which allows to conclude the proof of the lemma. ◀

To conclude the proof of Theorem 5, we recall that Algorithm 1 is sound and complete: S_m is the set of states that can be covered. Hence, from Lemma 6, we deduce that if $\mathsf{COVER}_{\mathtt{r}}(\mathcal{P}, F) \neq \emptyset$, then there is $\rho \in \mathsf{COVER}_{\mathtt{r}}(\mathcal{P}, F)$ such that:

1. $\mathsf{L}(\gamma) = S_m$;
2. $\#\mathsf{nodes}(\rho) = c(S_m) \leq |I| + 2m \leq |I| + 2(|Q| - |I|) = 2|Q| - |I|$;
3. $\max_{\mathsf{n}} \#\mathsf{steps}(\rho, \mathsf{n}) \leq m \leq |Q| - |I|$.

Therefore $\#\mathsf{steps}(\rho) \leq \big(\#\mathsf{nodes}(\rho)\big) \cdot \Big(\max_{\mathsf{n}} \#\mathsf{steps}(\rho, \mathsf{n})\Big) \leq 2|Q|^2$, so that we established the desired bounds for Theorem 5.

3.2 Upper bounds on cutoff and covering length for lossy networks

Perhaps surprisingly, Algorithm 1 also computes the set of states that can be covered by lossy executions. Concerning coverable states, the reconfigurable and lossy semantics thus agree. In Section 4, we will show that reconfigurable covering executions can be linearly more succinct than lossy covering executions.

▶ **Lemma 7.** *Algorithm 1 returns the set of coverable states for lossy broadcast networks. In particular,* $\mathsf{COVER}_{\mathtt{l}}(\mathcal{P}, F) \neq \emptyset$ *iff* $F \cap S \neq \emptyset$.

Indeed, we have $\mathsf{Exec}_{\mathtt{l}}(\mathcal{P}) \subseteq \mathsf{Exec}_{\mathtt{r}}(\mathcal{P})$. Therefore $\mathsf{COVER}_{\mathtt{l}}(\mathcal{P}, F) \neq \emptyset$ implies $\mathsf{COVER}_{\mathtt{r}}(\mathcal{P}, F) \neq \emptyset$ and by Lemma 4, we conclude $F \cap S \neq \emptyset$. The other direction of Lemma 7 is a consequence of the following theorem.

▶ **Theorem 8.** *Let* $\mathcal{P} = (Q, I, \Sigma, \Delta)$ *be a broadcast protocol, and* $F \subseteq Q$. *If* $S \cap F \neq \emptyset$, *then there exists* $\rho \in \mathsf{COVER}_{\mathtt{l}}(\mathcal{P}, F)$ *with* $\#\mathsf{nodes}(\rho) \leq 2|Q|$ *and* $\#\mathsf{steps}(\rho) \leq 2|Q|^2$.

Before going to the proof of Theorem 8, we show a copycat property for the lossy broadcast networks, as a counterpart of Proposition 2 for the lossy semantics. Since the communication topology is static in lossy networks, the following proposition explicitly relates the communication topologies in the initial execution and its copycat extension.

▶ **Proposition 9** (Copycat for lossy semantics). *Given* $\rho : \gamma_0 \to_1 \gamma_1 \cdots \to_1 \gamma_r$ *an execution, with* $\gamma_r = (\mathsf{N}, \mathsf{E}, \mathsf{L})$*, for every* $q \in \mathsf{L}(\gamma_r)$*, for every* $\mathsf{n}^q \in \mathsf{N}$ *such that* $\mathsf{L}(\mathsf{n}^q) = q$*, there exists* $s \in \mathbb{N}$ *and an execution* $\rho' : \gamma'_0 \to_1 \gamma'_1 \cdots \to_1 \gamma'_s$ *with* $\gamma'_s = (\mathsf{N}', \mathsf{E}', \mathsf{L}')$ *such that* $|\mathsf{N}'| = |\mathsf{N}|+1$*, there is an injection* $\iota : \mathsf{N} \to \mathsf{N}'$ *with for every* $\mathsf{n} \in \mathsf{N}$*,* $\mathsf{L}'(\iota(\mathsf{n})) = \mathsf{L}(\mathsf{n})$*, and for the extra node* $\mathsf{n}_{\mathsf{fresh}} \in \mathsf{N}' \setminus \iota(\mathsf{N})$*,* $\mathsf{L}'(\mathsf{n}_{\mathsf{fresh}}) = q$*, for every* $\mathsf{n} \in \mathsf{N}$*,* $\mathsf{n}_{\mathsf{fresh}} \sim' \iota(\mathsf{n})$ *iff* $\mathsf{n}^q \sim \mathsf{n}$*,* $\#\mathit{steps}(\rho', \mathsf{n}_{\mathsf{fresh}}) = \#\mathit{steps}(\rho, \mathsf{n}^q)$*, and* $\#\mathit{nonlost\_steps}(\rho', \mathsf{n}_{\mathsf{fresh}}) = 0$.

Proof. First notice that, from our definition of lossy semantics, the topology should be the same in γ_0 and in γ_r, hence we can write $\gamma_0 = (\mathsf{N}, \mathsf{E}, \mathsf{L}_0)$, and more generally, for every i, $\gamma_i = (\mathsf{N}, \mathsf{E}, \mathsf{L}_i)$. Define N' as a finite set such that $|\mathsf{N}'| = |\mathsf{N}| + 1$, and fix an injection $\iota : \mathsf{N} \to \mathsf{N}'$. Write $\mathsf{n}_{\mathsf{fresh}}$ for the unique element of $\mathsf{N}' \setminus \iota(\mathsf{N})$. Set $\mathsf{L}'_0(\iota(\mathsf{n})) = \mathsf{L}_0(\mathsf{n})$ for every $\mathsf{n} \in \mathsf{N}$, and $\mathsf{L}'_0(\mathsf{n}_{\mathsf{fresh}}) = \mathsf{L}_0(\mathsf{n}^q)$. Define the edge relation E' by its induced edge relation $\sim'$ such that $\iota(\mathsf{n}) \sim' \iota(\mathsf{n}')$ iff $\mathsf{n} \sim \mathsf{n}'$, and $\mathsf{n}_{\mathsf{fresh}} \sim' \iota(\mathsf{n}')$ iff $\mathsf{n}^q \sim \mathsf{n}'$.

The idea will then be to make $\mathsf{n}_{\mathsf{fresh}}$ follow what n^q is doing. Roughly, if n^q is receiving a message to progress, then we will connect $\mathsf{n}_{\mathsf{fresh}}$ to a relevant node to also receive the message; if n^q is broadcasting a message, then we will make $\mathsf{n}_{\mathsf{fresh}}$ broadcast a message and lose, so that no other node is impacted.

Formally, we will show by induction on i that for every $0 \leq i \leq r$, there is an execution $\rho'_i : \gamma'_0 \to_1 \gamma'_1 \cdots \to_1 \gamma'_{f(i)}$ for some $f(i)$, such that $\mathsf{L}'_i(\iota(\mathsf{n})) = \mathsf{L}_i(\mathsf{n})$ for every $\mathsf{n} \in \mathsf{N}$ and $\mathsf{L}'_i(\mathsf{n}_{\mathsf{fresh}}) = \mathsf{L}_i(\mathsf{n}^q)$. The initial case $i = 0$ is obvious. We then assume that we have constructed a relevant ρ'_i for some $i < r$, and we will extend it to ρ'_{i+1} as follows. We make a case distinction depending on the nature of the step $\gamma_i \to_1 \gamma_{i+1}$:

- Assume $\gamma_i \xrightarrow{\mathsf{n},!a}_1 \gamma_{i+1}$ is a broadcast message with $\mathsf{n}^q \neq \mathsf{n}$, then ρ'_{i+1} is obtained by extending ρ'_i with the broadcast $\gamma'_{f(i)} \xrightarrow{\iota(\mathsf{n}),!a} \gamma'_{f(i)+1}$, with the condition that it should be lost if and only if it was lost in the original execution. For checking correctness, we distinguish two cases:
  - the broadcast message was not lost, and $\mathsf{n}^q \sim \mathsf{n}$. Then, it is the case that $\mathsf{n}_{\mathsf{fresh}} \sim' \iota(\mathsf{n})$, hence $\mathsf{n}_{\mathsf{fresh}}$ also receives the message. By resolving properly the nondeterminism, we can make the label of $\mathsf{n}_{\mathsf{fresh}}$ become the same as the label of n^q in $\gamma'_{f(i)+1}$. Note also that all nodes in $\iota(\mathsf{N})$ can progress to the same states as those of N in γ_{i+1};
  - the broadcast message was lost, or $\mathsf{n}^q \not\sim \mathsf{n}$, then it is the case that the label of n^q has not been changed in $\gamma_i \xrightarrow{\mathsf{n},!a}_1 \gamma_{i+1}$, and so will the label of the fresh node in $\gamma'_{f(i)}$.
- Assume $\gamma_i \xrightarrow{\mathsf{n}^q,!a}_1 \gamma_{i+1}$ is a broadcast message, then we extend ρ'_i with the two steps $\gamma'_{f(i)} \xrightarrow{\iota(\mathsf{n}^q),!a} \gamma'_{f(i)+1} \xrightarrow{\mathsf{n}_{\mathsf{fresh}},!a} \gamma'_{f(i)+2}$ (resolving nondeterminism in a similar way as in $\gamma_i \xrightarrow{\mathsf{n}^q,!a}_1 \gamma_{i+1}$), and we make the last broadcast lossy whereas the broadcast from $\iota(\mathsf{n}^q)$ is lossy if and only if it was lossy in $\gamma_i \to_1 \gamma_{i+1}$.

This concludes the induction. Notice that in the constructed execution, node $\mathsf{n}_{\mathsf{fresh}}$ does not make any real sending. ◀

For any configuration $\gamma = (\mathsf{N}, \mathsf{E}, \mathsf{L})$ and a node n, we write $\mathsf{L}(\mathsf{n}) = \times$ if n is not important anymore in the execution, in other words all the required conditions in γ' such that $\gamma \xrightarrow{*}_1 \gamma'$ are still satisfied whatever $\mathsf{L}(\mathsf{n})$ is.

Recall the saturation algorithm and the ordering of the sets and the states: $S_0 = I, S_1, \ldots, S_m = S$ are the sets after each iteration and q_i is the state such that $q_i \in S_i \setminus S_{i-1}$ for all $1 \leq i \leq m$. We will refine the construction from the proof of Lemma 6 (in the context of reconfigurable broadcast networks), and build inductively a lossy execution covering all states in S_i. Since the topology is static, some nodes which have "finished their jobs" will remain connected to other nodes, and may therefore continue to change states (contrary to Lemma 6 where they could be fully disconnected). Hence, in every such execution, every state $q \in S_i$ (which is then covered by the execution) will have a main corresponding node, whose label will remain q. All nodes which are not the main node of a state will be assigned $\times$, since their labels will become meaningless.

We formalize this idea in the lemma below. However, for better understanding, we also illustrate this inductive construction of a witness execution in Figure 4 on the simple broadcast protocol from Figure 3. Configurations are represented vertically: they involve 10 nodes, and the communication topology is given for the first configuration only, for the sake of readability. To save space, several broadcasts (of the same message type, from different nodes) may happen in a *macrostep* that merges several steps. This is for instance the case in the first macrostep, where a is being broadcast from the node in set S_1, as well as from the first node in set S_2. Dashed arrows are used to represent that a node is not involved in some macrostep and thus stays in the same state. In the execution, the nodes that are performing a real broadcast are colored yellow, the ones which receive a message are colored gray, and blue nodes indicate the main nodes for the coverable states.

▶ **Lemma 10.** *For every step i of the refined saturation algorithm, there exists a configuration $\gamma = (\mathsf{N}, \mathsf{E}, \mathsf{L})$ and an execution $\rho : \gamma_0 \xrightarrow{*}_1 \gamma$ such that:*

- $\mathsf{L}(\gamma) \setminus \{\times\} = S_i$ *and* $\#\mathsf{nodes}(\rho) = c(S_i)$,
- $\max_{\mathsf{n}} \#\mathsf{steps}(\rho, \mathsf{n}) \leq i$ *and* $\max_{\mathsf{n}} \#\mathsf{nonlost\_steps}(\rho, \mathsf{n}) \leq 1$,
- *for every $q \in S_i$, there exists $\mathsf{n}_q^{\mathsf{main}} \in \mathsf{N}$ such that*
  - $\mathsf{L}(\mathsf{n}_q^{\mathsf{main}}) = q$ *and* $\#\mathsf{nonlost\_steps}(\rho, \mathsf{n}_q^{\mathsf{main}}) = 0$,
  - $\mathsf{n}_q^{\mathsf{main}} \sim \mathsf{n}$ *implies* $\mathsf{L}(\mathsf{n}) = \times$, *and if* $\mathsf{n} \notin \{\mathsf{n}_q^{\mathsf{main}} \mid q \in S_i\}$, *then* $\mathsf{L}(\mathsf{n}) = \times$.

Proof. We do the proof by induction on i. The case $i = 0$ is obvious, by picking one main node per initial state in I, and by disconnecting all nodes; hence forming an initial configuration satisfying all the requirements.

To prove the induction step, we distinguish two cases: depending on whether q_{i+1} was added as the target state of a broadcast action $!a$ from some $q \in S_i$; or whether q_{i+1} is the target state of a reception from some $q \in S_i$ with matching broadcast between two states already in S_i.

Case 1: There exists $q \in S_i$ with $q \xrightarrow{!a} q_{i+1}$. We apply the induction hypothesis to step i, and exhibit the various elements of the statement. Applying the copycat property for lossy broadcast systems (that is, Proposition 9) with node $\mathsf{n}_q^{\mathsf{main}}$, we build an execution $\rho' : \gamma'_0 \xrightarrow{*}_1 \gamma'$ such that $\gamma' = (\mathsf{N}, \mathsf{E}, \mathsf{L}')$ with $|\mathsf{N}'| = |\mathsf{N}| + 1$, and an appropriate injection ι. The fresh node $\mathsf{n}_{\mathsf{fresh}}$ is connected to nodes to which $\mathsf{n}_q^{\mathsf{main}}$ was connected before; hence, by induction hypothesis, it is only connected to nodes labelled with $\times$. Then we extend ρ' with $\gamma' \xrightarrow{\mathsf{n}_{\mathsf{fresh}}, !a} \gamma''$ and lose the message (this is for condition $\#\mathsf{nonlost\_steps}(\rho, \mathsf{n}_q^{\mathsf{main}}) = 0$ to be satisfied). We declare $\mathsf{n}_{q_{i+1}}^{\mathsf{main}} = \mathsf{n}_{\mathsf{fresh}}$. All requirements for γ'' are easily checked to be satisfied (when a node is labelled with $\times$ in γ', then it remains labelled by $\times$ in γ'').

Case 2: There exist $q, q', q'' \in S_i$ such that $q \xrightarrow{?a} q_{i+1}$ and $q' \xrightarrow{!a} q''$. We apply the induction hypothesis to step i, and exhibit the various elements of the statement. Applying twice the copycat property (that is, Proposition 9), once with node $\mathsf{n}_q^{\mathsf{main}}$ and once with

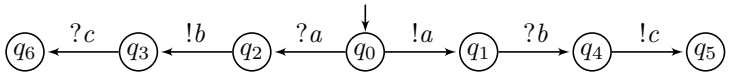

Figure 3 Illustrating example for the saturation algorithm.

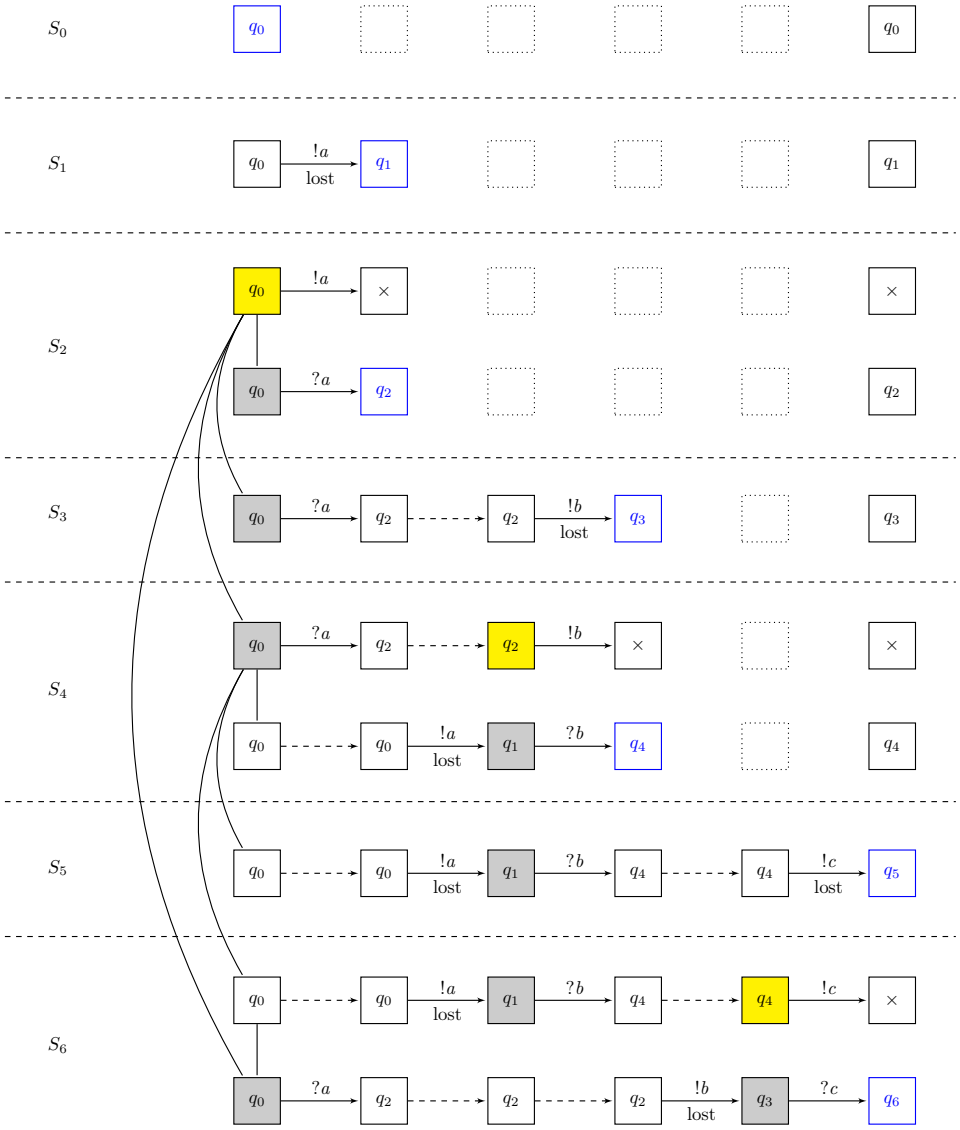

Figure 4 Applying saturation algorithm on protocol in Figure 3 in lossy semantics. Configurations are represented vertically; for readability, macrosteps merge several broadcasts.

node $\mathsf{n}_{q'}^{\mathsf{main}}$, we build an execution $\rho' : \gamma'_0 \xrightarrow{*}_1 \gamma'$ such that $\gamma' = (\mathsf{N}, \mathsf{E}, \mathsf{L}')$ with $|\mathsf{N}'| = |\mathsf{N}| + 2$, and an appropriate injection ι. The two fresh nodes $\mathsf{n}_{\mathsf{fresh}}$ and $\mathsf{n}'_{\mathsf{fresh}}$ are only connected to $\times$-nodes in γ' (by induction hypothesis on $\mathsf{n}_q^{\mathsf{main}}$ and $\mathsf{n}_{q'}^{\mathsf{main}}$ respectively). We transform γ'_0 into γ''_0 by connecting the two nodes $\mathsf{n}_{\mathsf{fresh}}$ and $\mathsf{n}'_{\mathsf{fresh}}$. By Proposition 9, we know that those two nodes don't perform any real sending (*i.e.*, $\#\mathsf{nonlost\_steps}(\rho', \mathsf{n}_{\mathsf{fresh}}) = 0$ and $\#\mathsf{nonlost\_steps}(\rho', \mathsf{n}'_{\mathsf{fresh}}) = 0$), hence this new connection will not affect the labels of the nodes, and we can safely apply the same transitions as in ρ' from γ''_0 to get an execution

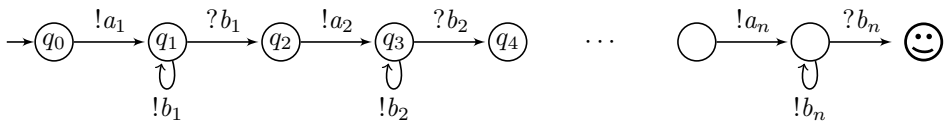

Figure 5 Broadcast protocol with linear cutoff and quadratic covering length.

$\rho'' : \gamma''_0 \xrightarrow{*}_1 \gamma''$, where γ'' coincides with γ', with an extra connection between nodes $\mathsf{n}_{\mathsf{fresh}}$ and $\mathsf{n}'_{\mathsf{fresh}}$. Then, we extend ρ'' with $\gamma'' \xrightarrow{\mathsf{n}'_{\mathsf{fresh}},!a} \gamma'''$. We assume it is a real sending, hence: node $\mathsf{n}_{\mathsf{fresh}}$ can progress from state q to q_{i+1}, and node $\mathsf{n}'_{\mathsf{fresh}}$ can progress from q' to q''. All other nodes which are connected to $\mathsf{n}'_{\mathsf{fresh}}$ are labelled by $\times$ in γ'', hence cannot be really affected by that sending. We relabel $\mathsf{n}'_{\mathsf{fresh}}$ to $\times$, and declare $\mathsf{n}^{\mathsf{main}}_{q_{i+1}} = \mathsf{n}_{\mathsf{fresh}}$. The expected conditions of the statement are easily checked to be satisfied by this new execution. ◀

Bounds are then obtained similarly to the reconfigurable case, see page 8.

3.3 Matching lower bounds for reconfigurable and lossy networks

In this section, we show that the linear bound on the cutoff and the quadratic bound on the length of witness executions are tight, both for the reconfigurable and the lossy broadcast networks.

▶ **Theorem 11.** *There exists a family of broadcast protocols $(\mathcal{P}_n)_n$ with $\mathcal{P}_n = (Q_n, I_n, \Sigma_n, \Delta_n)$, and target states $F_n \subseteq Q_n$ with $|Q_n| \in O(n)$, such that for every n, $\mathsf{COVER}_{\mathtt{r}}(\mathcal{P}_n, F_n) \neq \emptyset$, $\mathsf{COVER}_{\mathsf{l}}(\mathcal{P}_n, F_n) \neq \emptyset$, and any witness reconfigurable or lossy execution has size $O(n)$ and length $O(n^2)$.*

Proof. Consider $\mathcal{P}_n$, as depicted in Figure 5 with $2n{+}1$ states and $F_n = \{☺\}$. Any covering reconfigurable execution involves at least $n{+}1$ nodes, and has at least $\frac{n^2+5n}{2}$ steps. Indeed, intuitively, the process responsible for broadcasting b_i is blocked in q_{2i-1}, so that n such processes are needed, plus one process in ☺; moreover, $n{+}2{-}i$ broadcasts of a_i and one broadcast of each b_i happen. ◀

4 Succinctness of reconfigurations compared to losses

In this section, we show that reconfigurable executions can be linearly more succinct than lossy executions, in terms of number of nodes. Given the tight linear bound on cutoff, this is somehow optimal.

▶ **Theorem 12.** *There exists a family of broadcast protocols $(\mathcal{P}_n)_n$ with $\mathcal{P}_n = (Q_n, I_n, \Sigma_n, \Delta_n)$ and target states $F_n \subseteq Q_n$ such that for every n:*

- *there exists a reconfigurable covering execution in $\mathcal{P}_n$ with 3 nodes; and*
- *any lossy covering execution in $\mathcal{P}_n$ requires $O(n)$ nodes.*

Proof. $\mathcal{P}_n$ is depicted in Figure 6. It has $3n{+}2$ states and we let $F_n = \{☺\}$. A covering reconfigurable execution of size 3 is given in Figure 7. Colored nodes broadcast a message in the step leading to the next configuration. Along that execution, the top node always remains at q_0 and alternatively broadcasts a to the middle node and disconnects; the middle node follows the chain of q_i states and alternatively broadcasts b_i's to the bottom node which gradually progresses along the chain of states r_i and reaches ☺.

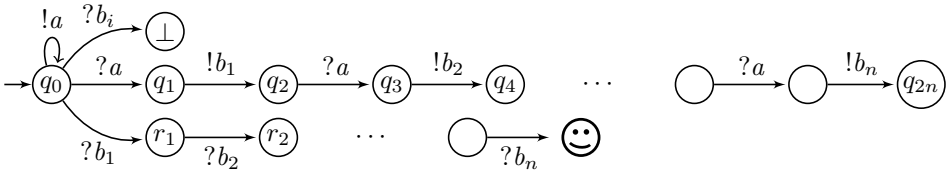

Figure 6 Example where reconfigurable semantics needs less nodes than lossy semantics.

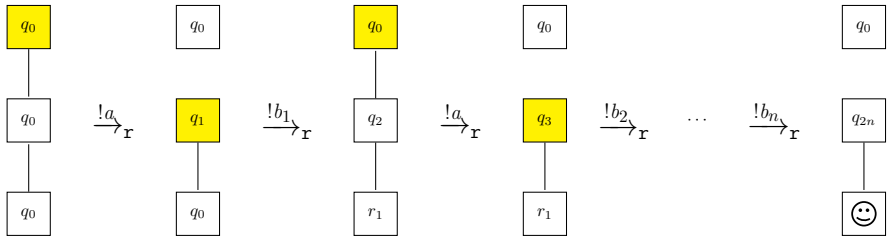

Figure 7 Covering reconfigurable execution with 3 nodes on the protocol from Figure 6.

Let us argue that in the lossy semantics, $O(n)$ nodes are needed to cover ☺. Obviously, one node, say $\mathsf{n}_{☺}$, is needed to reach the target state, after having received sequentially all the b_i's (which should then correspond to real broadcasts). Towards a contradiction, assume there is a node n which makes $\mathsf{n}_{☺}$ progress twice, that is, n is connected to $\mathsf{n}_{☺}$ and performs at least two real broadcasts, say $!b_i$ and $!b_j$ with $i < j$. Node n needs to receive $j - i > 0$ times the message a after the real $!b_i$ has occurred, hence there must be at least one node in state q_0 connected to n after the real $!b_i$ by n. This is not possible, since this node has received the real $!b_i$ while being in q_0, leading to $\bot$ if $i > 1$, otherwise $\bot$ or r_1. Hence, each broadcast $!b_i$ needs to be sent by a different node. This requires at least $n{+}1$ nodes, say $\{\mathsf{n}_i \mid 1 \leq i \leq n\} \cup \{\mathsf{n}_{☺}\}$: node n_i is responsible for broadcasting (with no loss) b_i and $\mathsf{n}_{☺}$ progresses towards ☺. Notice that $\mathsf{n}_{☺}$ might be the node responsible for broadcasting all the a's. We conclude that $n{+}1$ is a lower bound on the number of nodes needed to cover ☺ in the lossy semantics.

To complete this example, observe that $n{+}1$ nodes do actually suffice in lossy semantics to cover ☺. Let $\mathsf{N} = \{\mathsf{n}_i \mid 1 \leq i \leq n\} \cup \{\mathsf{n}_{☺}\}$ and consider the static communication topology defined by $\mathsf{n}_i \sim \mathsf{n}_{☺}$ for every i. In the covering lossy execution, node $\mathsf{n}_{☺}$ initially broadcasts a's, so that all its neighbours, the n_i's can move to q_{2i-1}, using lost sendings. Then the each node n_i broadcasts its message $b_i$to $\mathsf{n}_{☺}$, starting with n_1 until n_n, so that $\mathsf{n}_{☺}$ reaches ☺. ◀

5 Complexity of deciding the size of minimal witnesses

We now consider the following decision problem of determining the minimal size of coverability witnesses for both the reconfigurable and lossy semantics.

MINIMUM NUMBER OF NODES FOR COVERABILITY (MINCOVER)
Input: A broadcast protocol $\mathcal{P}$, a set of states $F \subseteq Q$, and $k \in \mathbb{N}$.
Question: Does there exist a reconfigurable/lossy execution ρ covering some state in F, and with $\#\mathsf{nodes}(\rho) = k$?

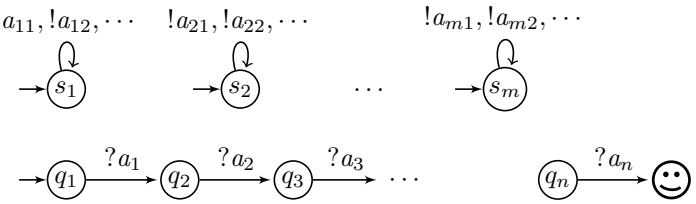

Figure 8 Illustration of the reduction to prove NP-hardness of MinCover.

By the copycat properties (for both semantics), if there is a covering execution of size less than k, then there is one of size exactly k.

▶ **Theorem 13.** MinCover *is* NP*-complete for both reconfigurable and lossy broadcast networks.*

The NP-hardness of MinCover is proved by reduction from SetCover, which is known to be NP-complete [9]. Recall that SetCover takes as input a finite set of elements $\mathcal{U}$, a collection $\mathcal{S}$ of subsets of $\mathcal{U}$ and an integer k, and returns yes iff there exists a subcollection of $\mathcal{S}$ of size at most k that covers $\mathcal{U}$.

Given an instance of the SetCover problem $(\mathcal{U}, \mathcal{S}, k)$ with $\mathcal{U} = \{a_1, a_2, \ldots, a_n\}$ and $\mathcal{S} = \{S_1, S_2, \ldots, S_m\}$, we build a protocol $\mathcal{P} = (Q, I, \Sigma, \Delta)$ as depicted in Figure 8, where we assume $S_i = \{a_{i1}, a_{i2}, \ldots\}$ for every i.

We can then show that $\mathcal{U}$ has a cover using $\mathcal{S}$ of size k if and only if there exists a reconfigurable/lossy execution for $\mathcal{P}$ covering F and with $k{+}1$ nodes.

For the NP-membership, it suffices to observe that the length of a minimal covering execution is polynomially bounded, thanks to Theorem 5 and 8. Moreover, configurations and updates of configurations by given transitions can be represented in and computed in a compact way. It is thus possible to implement a guess-and-check NP-algorithm for the MinCover problem, that non deterministically guesses an execution with k nodes of maximal length that is polynomially bounded in the size of the broadcast protocol.

6 Conclusion

In this paper, we have given a tight linear bound on the cutoff and a tight quadratic bound on the covering length for reconfigurable broadcast networks. We have also proposed a new semantics for broadcast networks with a static topology, where messages can be lost at sending. Similar tight bounds can be proven for that new semantics. Proofs are based on a refinement of the saturation algorithm of [2], and on fine analysis of copycat lemmas. As a side result of these constructions, we get that the set of states which can be covered by the two semantics is actually the same, but that the reconfigurable semantics can be linearly more succinct (in terms of number of nodes). We also prove the NP-completeness for the existence of a witness execution with the minimal number of nodes.

As future work, we want to pursue the study of the model with stochastic losses, and design analysis algorithms for various quantitative questions. Also, in this work we have not studied the tradeoff between number of nodes and length of covering computation. The precise interplay between number of nodes and length of covering execution is a possible direction for future work.

References

1. Roderick Bloem, Swen Jacobs, Ayrat Khalimov, Igor Konnov, Sasha Rubin, Helmut Veith, and Josef Widder. *Decidability of Parameterized Verification*. Synthesis Lectures on Distributed Computing Theory. Morgan & Claypool Publishers, 2015. doi:10.2200/S00658ED1V01Y201508DCT013.
2. Giorgio Delzanno, Arnaud Sangnier, Riccardo Traverso, and Gianluigi Zavattaro. On the complexity of parameterized reachability in reconfigurable broadcast networks. In *Proceedings of the 32nd Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'12)*, volume 18 of *Leibniz International Proceedings in Informatics*, pages 289–300. Leibniz-Zentrum für Informatik, December 2012. doi:10.4230/LIPIcs.FSTTCS.2012.289.
3. Giorgio Delzanno, Arnaud Sangnier, and Gianluigi Zavattaro. Parameterized verification of ad hoc networks. In *Proceedings of the 21st International Conference on Concurrency Theory (CONCUR'10)*, volume 6269 of *Lecture Notes in Computer Science*, pages 313–327. Springer, September 2010. doi:10.1007/978-3-642-15375-4_22.
4. Giorgio Delzanno, Arnaud Sangnier, and Gianluigi Zavattaro. Verification of ad hoc networks with node and communication failures. In *Proceedings of the 32nd International Conference on Formal Techniques for Distributed Systems (FMOODS/FORTE'12)*, volume 7273 of *Lecture Notes in Computer Science*. Springer, 2012.
5. E. Allen Emerson and A. Prasad Sistla. Symmetry and model checking. *Formal Methods in System Design*, 9(1-2):105–131, August 1996. doi:10.1007/BF00625970.
6. Javier Esparza. Keeping a crowd safe: On the complexity of parameterized verification (invited talk). In *Proceedings of the 31st Symposium on Theoretical Aspects of Computer Science (STACS'14)*, volume 25 of *Leibniz International Proceedings in Informatics*, pages 1–10. Leibniz-Zentrum für Informatik, March 2014. doi:10.4230/LIPIcs.STACS.2014.1.
7. Javier Esparza, Pierre Ganty, and Rupak Majumdar. Parameterized verification of asynchronous shared-memory systems. In *Proceedings of the 25th International Conference on Computer Aided Verification (CAV'13)*, volume 8044 of *Lecture Notes in Computer Science*, pages 124–140. Springer, July 2013. doi:10.1007/978-3-642-39799-8_8.
8. Steven M. German and A. Prasad Sistla. Reasoning about systems with many processes. *Journal of the ACM*, 39(3):675–735, July 1992. doi:10.1145/146637.146681.
9. Richard M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, The IBM Research Symposia Series, pages 85–103, 1972.
10. Marvin Minsky. *Computation: Finite and Infinite Machines*. Prentice Hall International, 1967.
11. Ichiro Suzuki. Proving properties of a ring of finite-state machines. *Information Processing Letters*, 28(4):213–214, 1988. doi:10.1016/0020-0190(88)90211-6.